AMENDMENT TO THE SPECIFICATION

Please amend the specification to be consistent with the proposed changes to Figs. 1-3 as follows:

On page 4, please amend the paragraph starting on line 13 as follows:

FIGs. ~~2 and 3~~ 2, 2A, 2B, 3, and 3A are flow charts depicting the operations performed by the arrangement depicted in FIG. 1, which is used in understanding the invention, with ~~FIG. 2~~ FIGs. 2-2B depicting operations performed in connection with obtaining and dynamic loading of the stub information and ~~FIG. 3~~ FIGs. 3 and 3A depicting operations performed in connection with use of the stub information to invoke processing of the remote method or procedure.

On page 13, please amend the paragraph starting on line 10 as follows:

With this background, the operations performed by client computer 11(n), server computer 12(m) and, if necessary, nameserver 13 in connection with obtaining and dynamic loading of a stub class instance when a reference to a remote method is received will be described in connection with the flow chart depicted in ~~FIG. 2~~ FIGs. 2-2B. In addition, operations performed by the client computer 11(n) and server computer 12(m) in connection with remote invocation of a method using the stub class instance will be described in connection with the flow chart depicted in ~~FIG. 3~~ FIGs. 3 and 3A. With reference initially to FIG. 2, the execution environment control module 19 will, when it receives a reference to a remote method, initially determine whether an appropriate stub class instance is present in the execution environment 20 to facilitate invocation of the remote method (step 100). If the control module 19 determines that such a stub class instance 30 for the remote method is present in the execution environment, it may continue other operations (step 101). However, if the control module 19 determines in step 101 that such a stub class instance is not present in the execution environment 20 for the remote method, the control module 19 will use the stub class loader 33 to attempt to locate and load a stub class instance 30 for the class to process the remote method. In that case, the control module 19 will initially determine whether the invocation from the class instance 22 included a resource locator to identify the server computer 12(m) or other resource which maintains the class for the method to be invoked, or whether it (that is, the control module 19) or the stub class loader 33 otherwise are provided with such a resource locator (step 102). If the control module 19 makes a positive determination in that step, it will sequence to step 103 to enable the stub class loader 33 to initiate communications with identified server computer 12(m) to obtain stub class instance for the class and method to be invoked (step 103). When the stub class loader 33 receives the stub class instance 30 from the server computer 12(m), it will load the stub class instance 30 into execution environment 20 for the class instance 22 which initiated the remote method invocation call in step 100 (step 104). After the stub class instance 30 for the referenced remote method has been loaded in the execution environment, the method can be invoked as will be described below in connection with ~~FIG. 3~~ FIGs. 3 and 3A.

On page 14, please amend the paragraph starting on line 16 as follows:

Alternatively, the control module 19 may attempt to obtain a resource locator from the narneserver computer 13 or other resource provided by the network 14 (generally represented in FIG. 1 by the narneserver computer 13), using a call to, for example, a default stub class instance 30. The call to the default stub class instance 30 will include an identification of the class and method to be invoked and the name of the nameserver computer 13(m). Using the default stub class instance 30, the control module 19 will enable the computer 11(n) to initiate communications with nameserver computer 13 to obtain an identifier for a server computer 12(m) which maintains the class and method to be invoked (step 110), as shown in FIG. 2A. The communications from the default stub class instance 30 will essentially correspond to a remote method invocation, with the method enabling the nameserver computer to provide the identification for the server computer 12(m), if one exists associated with the class and method to be remotely invoked, or alternatively to provide an indication that no server computer 12(m) is identified as being associated with the class and method. During the communications in step 110, the default stub class interface 30 will provide, as a parameter value, the identification of class method to be invoked.

On page 15, please amend the paragraph starting on line 10 as follows:

After receipt of the result information from the nameserver computer 13, the default stub class instance, under control of the control module 19, will pass result information to the stub class loader 33 (step 113), as shown in FIG. 2B. Thereafter, the stub class loader 33 determines whether the result information from the nameserver computer comprises the identification for the server computer 12(m) or an indication that no server computer 12(m) is identified as being associated with the class (step 114). If the stub class loader 33 determines that the result information comprises the identification for the server computer 12(m), it (that is, the stub class loader 33) will return to step 103 to initiate communication with the identified server computer 12(m) to obtain stub class instance for the class and method that may be invoked. On the other hand, if the stub class loader 33 determines in step 114 that the nameserver computer 13 had provided an indication that no server computer 12(m) is identified as being associated with the class and method that may be invoked, the "class not found" exception may be indicated (step 115) and an exception handler called as described above.

On page 15, please amend the paragraph starting on line 22 as follows:

As noted above, the stub class instance 30 retrieved and loaded as described above in connection with ~~FIG. 2~~ FIGs. 2-2B may be used in remote invocation of the method. Operations performed by the client computer 11(n) in connection with remote invocation of the method will be described in connection with the ~~flow chart in FIG. 3~~ flow charts in FIGs. 3 and 3A. As depicted in ~~FIG. 3~~ FIGs. 3 and 3A, when a class instance 22 invokes a method, the control module 19 may initially verify that a stub class instance 30 is present in the execution environment for remote method to be invoked (step 120). If a positive determination is made in step 120, the stub class instance 30 will be used for the remote invocation, and in the remote invocation will provide parameter values which are to be used in processing the remote method (step 121). Thereafter, the stub class instance 30 for the remote method that may be invoked will be used to initiate communications with the server computer 12(m) which maintains the class for the remote method (step 122), in the process, the parameter values which are to be used in processing the remote method will be passed. It will be appreciated that, if the server computer 12(m) which is to process the method is the same physical computer as the client computer 11(n) which is invoking the method, the communications can be among execution environments which are being processed within the physical computer. On the other hand, if the server computer 12(m) which is to process the method is a different physical computer from that of the client computer 11(n) which is invoking the method, the communications will be through the client computer's and server computer's respective network interfaces 15(n) and 16(m) and over the network 14.

On page 16, please amend the paragraph starting on line 13 as follows:

In response to the communications from the stub class instance in step 122, the server computer 12(m), if necessary establishes an execution environment 24 for the class which maintains the method that may be invoked, and the uses the information provided by the skeleton 32 to create a class instance 26 for that class (step 123). Thereafter, the server computer 12(m), under control of the control module 28, will process the method in connection with parameter values that were provided by stub class instance 30 (step 124), as shown in FIG. 3A. After completing processing of the method, the server computer 12(m), also under control of the control module 28, will initiate communications with the client computer's stub class instance 30 to provide result information to the stub class instance (step 125). In a manner similar to that described above in connection with step 102, if the server computer 12(m) which processed the method is the same physical computer as the client computer 11(n) which invoked the method, the communications can be among execution environments 24 and 20 which are being processed within the physical computer. On the other hand, if the server computer 12(m) which processed the method is a different physical computer from that of the client computer 11(n) which is invoking the method, the communications will be through the server computer's and client computer's respective network interfaces 16(m) and 15(n) and over the network 14. After the stub class instance 30 receives the result information from the server computer, it may provide result information to the class instance 22 which initiated the remote method invocation (step 126), and that class instance 22 can continue processing under control of the control module 19.